CSC110 Fall 2021: Term Test 2
Question 2 (Analyzing Algorithm Running Time)

Jamie Yi

Wednesday December 8, 2021

Question 2, Part 1

We define the function $g : \mathbb{N} \to \mathbb{R}^{\geq 0}$ as $g(n) = 5n^2(n-2)$. Consider the following statement:

$$g(n) \in \mathcal{O}(n^4)$$

(a) Rewrite the statement $g(n) \in \mathcal{O}(n^4)$ by expanding the definition of Big-O.

Solution:
$\exists c, n_0 \in \mathbb{R}^+$ s.t. $\forall n \in \mathbb{R},\ n \geq n_0 \implies g(n) \leq c \cdot n^4$

(b) Write the *negation* of the statement from (a), using negation rules to simplify the statement as much as possible.

Solution:
$\forall c, n_0 \in \mathbb{R}^+$, $\exists n \in \mathbb{R}$ s.t. $n \geq n_0 \land g(n) > c \cdot n^4$

(c) Which of statements (a) and (b) is true? Provide a complete proof that justifies your choice.
In your proof, you may not use any properties or theorems about Big-O/Omega/Theta. Work from the expanded statement from (a) or (b).

Solution:

Proof. Let $n_0 = 1$
Let $c = 5$
Prove that $\forall c, n_0 \in \mathbb{R}^+$, $\exists n \in \mathbb{R}$ s.t. $n \geq n_0 \land g(n) > c \cdot n^4$
We know, $5 \leq 5n$ (since by definition of Big-O $n \geq 1$)
Then, $5n^3 \leq 5n^4$
Then, $5n^3 - 10n^2 \leq 5n^3 \leq 5n^4$ ($10n^2$ is always positive)
Then, $5n^3 - 10n^2 \leq 5n^4$
Thus, $5n^2(n-2) \leq c \cdot n^4$ as needed

□

Question 2, Part 2

Consider the function below.

```
def f(nums: list[int]) -> list[int]:          # Line 1
    n = len(nums)                             # Line 2
    i = 1                                     # Line 3
    new_list = []                             # Line 4
    while i < n:                              # Line 5
        if nums[i] % 5 == 0:                  # Line 6
            new_list = [i * j for j in nums]  # Line 7
        else:                                 # Line 8
            list.append(new_list, i)          # Line 9
        i = i * 4                             # Line 10
    return new_list                           # Line 11
```

(a) Perform an *upper bound analysis* on the worst-case running time of `f`. The Big-O expression that you conclude should be *tight*, meaning that the worst-case running time should be Theta of this expression, but you are not required to show that here.

To simplify your analysis, you may omit all floors and ceilings in your calculations (if applicable). Use "at most" or $\leq$ to be explicit about where a step count expression is an upper bound.

Solution:

Treat the first three statements as a block of constant time statements, I will count them as 1 step

While loop analysis:

-let k be the iteration

-let i_k be the value of i at iteration k

-$i_k = 4k$

-the loop terminates when $i_k \geq n$

-$4k \geq n$

-$k \geq \frac{n}{4}$

-So the loop will iterate at most k times, where k is the closest integer to $\frac{n}{4}$, rounded up, or more specifically $\lceil \frac{n}{4} \rceil$ times

Of the two possible if-else branches, the first if branch takes more steps, I will count it as taking at most n steps

The last return statement is constant time, and takes 1 step.

Thus 2 constant time steps are taken outside the loop, and the loop itself will take at most $n\lceil \frac{n}{4} \rceil$ steps

So the total runtime of the function is $RC = n\lceil \frac{n}{4} \rceil + 2 \in \mathcal{O}(n^2)$

(b) Perform a *lower bound analysis* on the worst-case running time of `f`. The Omega expression you find should match your Big-O expression from part (a).

Hint: you don't need to try to find an "exact maximum running-time" input. *Any* input family whose running time is Omega of ("at least") the bound you found in part (a) will yield a correct analysis for this part.

Solution:

Let $n \in \mathbb{N}$, let nums be a list of length n containing only integers that 5 divides, that is, all of the integers in nums are multiples of 5

In this case the first if condition in the while loop will trigger every time, taking n steps per iteration of the loop(since every element of nums is cleanly divisible by 5).

The loop itself will always iterate $\lceil \frac{n}{4} \rceil$ times, no matter what is contained in nums(only the length of nums matters)

The statements outside the loop will always also be constant time no matter what.

Thus the total number of steps taken is $n\lceil \frac{n}{4} \rceil + 2$, which is $\mathcal{O}(n^2)$

SUBMIT THIS FILE AND THE GENERATED PDF q2.pdf FOR GRADING